Journal of Sustainability, Policy, and Practice

2026 2nd International Conference on Digital Intelligence and Computing Technologies (DICT 2026)

Article

Temporal Deep Learning for Financial Fraud Detection: A Comparative Evaluation of LSTM, GRU, Transformer, and TCN on Real Transaction Datasets

Yinzhe Lu[1,*]

[1] Business Analytics, Fordham University, NY, USA
* Correspondence: Yinzhe Lu, Business Analytics, Fordham University, NY, USA

Abstract: Financial anomaly detection has evolved from static classification toward sequence-aware learning that leverages the temporal structure of transaction streams. Four families of temporal neural architectures are now routinely applied in this domain: Long Short-Term Memory (LSTM) networks, Gated Recurrent Units (GRU), Transformer encoders, and Temporal Convolutional Networks (TCN). Existing empirical comparisons are typically restricted to a single dataset and a single problem granularity, leaving open the question of how these architectures rank when evaluated under a common protocol across transaction-level and node-level risk prediction. This paper presents a controlled empirical comparison of the four architectures on three real financial datasets: the ULB European credit-card transactions released by Worldline and the ULB Machine Learning Group, the IEEE-CIS card-not-present dataset contributed by Vesta Corporation, and the Elliptic Bitcoin transaction graph released by Elliptic together with the MIT-IBM Watson AI Lab. Under a unified sequence-learning interface with matched preprocessing and matched parameter budgets, the four architectures are examined using ROC-AUC, PR-AUC, F1, and recall. The Transformer attains the highest mean ROC-AUC on each of the three datasets, with GRU and TCN close behind and LSTM slightly trailing; on the smallest dataset, however, the spread is of the same order as the five-seed standard deviation, and the cross-dataset ordering should be read as an ordering of sample means under a shared sequence interface rather than as a strict apples-to-apples separation. The widest inter-architecture F1 gap emerges on the Elliptic node-level setting. By framing anomaly detection as a temporal behavioral modeling problem rather than an isolated classification task, the study highlights the capacity of sequence-aware architectures to capture evolving transaction patterns and positions temporal scoring as a mechanism to strengthen platform reliability and user trust.

Keywords: temporal deep learning; financial anomaly detection; empirical evaluation; transaction sequence modeling; complex system behavioral modeling; platform reliability

Received: 26 April 2026
Revised: 16 June 2026
Accepted: 29 June 2026
Published: 03 July 2026

1. Introduction

1.1. Background and Motivation

Digital commerce and real-time payment platforms process billions of transactions every day, and the cost of undetected fraud continues to rise with their volume. Operational risk teams now treat fraud detection as a streaming classification problem in which each transaction must be scored within tens of milliseconds, and the relevant evidence is seldom confined to a single event. Customer spending habits, merchant profiles, device fingerprints, and adversary tactics all drift over time, producing concept drift on top of the severe class imbalance inherent in fraud data, and together making

traditional tabular classifiers fragile in production settings [1]. A practical fraud-scoring pipeline also has to contend with verification latency, because the ground-truth label for a transaction is confirmed only after cardholders raise disputes, while the scoring engine must continue issuing predictions in real time. The combination of these factors --- high volume, drifting distributions, class imbalance of 0.1% to 3%, and delayed labels --- sets fraud detection apart from ordinary supervised classification tasks and has driven a steady migration toward models that reason about events in context.

Viewed through a broader lens, fraud detection in modern digital financial platforms is not merely a per-transaction binary classification problem but also a temporal behavioral modeling challenge in which risk unfolds across evolving transaction histories. Fraudulent events often emerge as deviations from normal sequential patterns rather than as anomalies in any single field, which makes sequence-aware modeling particularly relevant in large-scale payment environments. From this perspective, temporal deep learning architectures are not only predicting labels but also learning recurring and diverging behavioral patterns over time, a framing that aligns the technical contribution with the operational reality of high-volume digital transaction platforms.

These difficulties motivated a shift from static per-transaction classifiers toward sequence-aware learners that ingest a window of prior events together with the current one. Sequence learners exploit the observation that fraudulent cards or accounts typically exhibit atypical temporal footprints, such as a sudden burst of small trial charges, an unusual cadence of geographically dispersed transactions, or a rapid escalation toward high-value purchases. Jurgovsky et al. demonstrated, using industrial card-not-present data, that formulating fraud detection as sequence classification and training an LSTM on rolling transaction windows improves over static tree ensembles on offline point-of-sale traffic, while also reshaping which frauds are caught [2]. That finding has since been reproduced across a wide range of deep sequential architectures, yet different studies use different datasets, sequence lengths, and evaluation metrics, making it hard to judge which temporal inductive bias is best suited to fraud detection in operational deployment. A broader review of deep learning for time-series classification across application domains also reports that no single sequence architecture dominates across tasks, which reinforces the need for domain-specific empirical audits rather than blanket architectural recommendations [3].

A further consideration motivating the present comparative study is the growing practical importance of deep temporal models in the high-dimensional, multi-source data regime characterizing modern financial platforms. Transaction streams in production environments carry hundreds of engineered features drawn from payment networks, device-fingerprinting services, geolocation providers, and behavioral-analytics pipelines, and the discriminative signal for fraud often resides in cross-system temporal correlations that span tens or hundreds of prior events. Traditional rule engines and simple statistical thresholds, while interpretable and fast to deploy, are fundamentally limited in their capacity to capture these long-range, cross-subsystem dependencies. Sequence-aware deep learning architectures, by contrast, can learn hierarchical temporal representations that distill complex feature interactions over extended histories, enabling the scoring engine to detect subtle drift patterns well before they surface as overt fraud. This capacity for learning long-term dependencies from high-dimensional temporal data is what distinguishes deep sequential models from their shallow predecessors and underpins the practical relevance of the architectural comparison conducted in this paper.

1.2. Contributions and Paper Organization

1.2.1. Scope of the Comparative Study

This paper compares four representative temporal architectures --- LSTM, GRU, Transformer encoder, and TCN --- and evaluates each on three publicly auditable, real-world financial datasets that jointly cover transaction-level fraud, card-not-present e-commerce fraud, and node-level illicit-transaction detection on a cryptocurrency ledger. All four architectures are trained and evaluated under a unified sequence-learning

interface, with matched preprocessing where data formats allow, matched parameter and epoch budgets, and a temporal split enforced per dataset to control the most common leakage pathways. The study deliberately avoids proposing a new architecture and instead presents a disciplined audit of what the existing temporal toolkit delivers when applied to operational fraud data. Doing so narrows --- rather than fully eliminates --- the effect of the inductive bias relative to confounders such as dataset choice, feature engineering, and loss-function tuning, which have been the main sources of inconsistent conclusions in the prior literature. The three datasets are not a homogeneous benchmark in the strict sense: ULB and IEEE-CIS are cast as per-transaction sequence classification, whereas the Elliptic setting is recast as a node-level sequence-classification problem over a transaction graph, and the resulting comparison is best understood as a comparison under a shared sequence interface rather than as a native apples-to-apples test.

1.2.2. Contributions

The contributions of this work are threefold. A reproducible empirical protocol is defined that aligns transaction-level and node-level fraud detection under a shared sequence-learning interface, so that the four architectures are evaluated on a common evaluation surface rather than on dataset-specific pipelines. A consistent set of ROC-AUC, PR-AUC, F1, and recall measurements is then reported across the three datasets, quantifying how the four architectures compare under both balanced and highly imbalanced traffic. A sensitivity analysis on the input sequence length is finally provided, which surfaces the regimes in which each architecture has a measurable edge and reveals how much each architecture actually exploits a longer history. The remainder of the paper is organized as follows. Section 2 surveys prior work on recurrent, attention-based, and convolutional temporal encoders for fraud detection. Section 3 describes the datasets, preprocessing, and training protocol. Section 4 reports the empirical results and the sensitivity analyses. Section 5 discusses the implications for operational risk monitoring and outlines the study's limitations.

2. Related Work

2.1. Recurrent Architectures for Fraud Detection

2.1.1. LSTM-Based Methods

Long Short-Term Memory networks use gated memory cells to propagate information across long sequences without the vanishing-gradient pathologies of vanilla recurrent networks, which made them the workhorse of sequence modeling in the deep-learning era [4]. They became the default sequence model for fraud detection because they naturally encode the rolling transaction history associated with a card or account, and because the gating mechanism is flexible enough to memorize rare but diagnostic past events such as a device fingerprint change or a country jump. Recent work has augmented the recurrent backbone with attention to further sharpen which past transactions matter for the current decision. Benchaji et al. combine a UMAP-based feature selector with an attention-gated LSTM on the ULB European card dataset and on a Brazilian card dataset, and report improvements in precision-recall area that they attribute to the model's ability to up-weight the most diagnostic transactions in each cardholder window [5]. Sequential LSTMs have also been used as a feature-extraction stage within hybrid architectures, where the recurrent features feed into a downstream classifier alongside engineered aggregates.

2.1.2. GRU-Based and Ensemble Approaches

Gated Recurrent Units were proposed as part of encoder--decoder machine translation architectures and merge the input and forget gates of an LSTM into a single update gate, while collapsing the cell state into the hidden state, yielding a lighter model with fewer parameters while maintaining similar expressive power [6]. In fraud detection, this parameter economy has been exploited by Forough and Momtazi, who build an ensemble of deep sequential models in which several GRU (and in the ablation several

LSTM) base learners vote through a feed-forward meta-classifier, and report that the GRU ensemble matches or exceeds the LSTM ensemble on the European and Brazilian card datasets while training substantially faster [7]. The relative compactness of GRU is particularly attractive in streaming fraud scoring, where per-event inference latency is a hard constraint, and where retraining windows of a few hours imply that training-time efficiency translates directly into lower end-to-end operational cost.

2.2. Attention and Convolutional Architectures for Transaction Sequences

Parallel to the recurrent line, two non-recurrent families have gained traction. The Transformer, originally formulated for machine translation, replaces recurrence with multi-head self-attention and delivers longer effective context at a lower sequential compute cost, making it well-suited to long transaction histories and to parallel training on modern accelerators [8]. On fraud data, Cheng et al. combine temporal attention with 3D convolutions to form STAN, an attention-based detector that outperforms both static baselines and a plain LSTM on a commercial card-not-present stream [9]. Xiang et al. generalize this line to semi-supervised learning on a temporal transaction graph using a gated temporal attention network, showing that attention to neighboring histories recovers fraud patterns that escape purely sequential detectors [10]. Temporal Convolutional Networks use dilated causal convolutions to aggregate long histories in parallel while preserving the temporal ordering and were shown across a broad benchmark of sequence-modeling tasks to match or exceed LSTM in accuracy while demonstrating longer effective memory and fully parallelizable training, which has motivated their adoption as a lightweight alternative to recurrent encoders in streaming applications [11].

3. Experimental Setup

3.1. Datasets

Three real-world financial datasets are adopted, chosen to span different granularities, scales of class imbalance, and payment technologies. The ULB Credit Card Fraud dataset contains 284,807 credit-card transactions made by European cardholders across two days in September 2013, of which 492 are labeled as fraudulent, yielding an imbalance ratio of 0.172%; confidentiality constraints mean that the 28 anonymized features V1 through V28 are PCA projections, with only the Time and Amount fields retained in raw form, and the data were released by Worldline together with the ULB Machine Learning Group under the Database Contents License v1.0. The IEEE-CIS Fraud Detection dataset, contributed by Vesta Corporation for the IEEE Computational Intelligence Society competition hosted on Kaggle, contains 590,540 training-stage card-not-present transactions with 394 transaction columns and 41 identity columns, of which 20,663 are labeled as fraud, corresponding to a 3.50% fraud rate; the transaction timeline is anonymized and spans roughly six months relative to a reference datetime. The Elliptic Bitcoin dataset is a time-series graph in which 203,769 Bitcoin transactions form the nodes and 234,355 directed payment flows form the edges, with 166 features per node (94 local plus 72 one-hop aggregated) and 49 discrete time steps; of the labeled subset, 4,545 transaction nodes are illicit (2% of all nodes) and 42,019 are licit (21%), while the remainder are unlabelled [12]. Consistent with the official Elliptic release, we refer to the labeled entities throughout the paper as transaction nodes rather than accounts. Table 1 summarizes the three datasets side by side; all values are taken from the official Kaggle pages, the companion paper, and the released documentation.

Table 1. Specifications of the Three Real Financial Datasets Used in the Experiments.

Dataset	Source	Records	Positive	Positive rate	Features	Time span	License
ULB Credit Card	Worldline & ULB ML Group (Kaggle: mlg-ulb/creditcardfraud)	284,807 transactions	492	0.172%	30 (V1–V28 PCA + Time + Amount)	2 consecutive days, Sep 2013	DbCL v1.0
IEEE-CIS	Vesta Corporation (Kaggle: ieee-fraud-detection)	590,540 training transactions	20,663	3.50%	394 transaction + 41 identity columns	~6 months (anonymized)	Kaggle competition terms (non-commercial research)
Elliptic Bitcoin	Elliptic & MIT-IBM Watson AI Lab (Kaggle: ellipticco/elliptic-dataset)	203,769 transaction nodes / 234,355 directed edges	4,545 illicit nodes (of 46,564 labeled)	2.00% of all nodes / 9.76% of labeled	166 per node (94 local + 72 one-hop aggregated)	49 discrete time steps (~2 years)	Public research release (non-commercial)

3.2. Data Preprocessing and Sequence Construction

Preprocessing is kept as close to identical as the three data formats allow, subject to the unavoidable structural differences between transaction-level streams and the Elliptic time-indexed graph. Numerical features are standardized using statistics fitted on the training split only to prevent leakage. In the ULB set, Amount is first transformed with a log(1+x) mapping to reduce skewness and then standardized along with the remaining numerical inputs; Time is retained as a continuous elapsed-time feature and standardized without an additional log transform. For IEEE-CIS, categorical columns are target-encoded on the training split only, with both the category-to-target mapping and the prior-smoothing statistics frozen before being applied to validation and test; identity features missing for more than 90% of rows are dropped before encoding, which removes 10 of the 41 identity columns and leaves 31 retained identity features, and remaining missing values in the retained identity columns are imputed using training-fold medians for numerical fields and training-fold category frequencies for categorical fields. Temporal splits are enforced throughout: on ULB and IEEE-CIS, the last 20% of the transaction

timeline is reserved for testing and the preceding 20% for validation, and on Elliptic, the canonical split at time step 34 is adopted, with steps 1--34 forming the training and validation windows and steps 35--49 forming the held-out test window.

Beyond the temporal boundary, four specific leakage pathways are controlled explicitly. First, standardization statistics, target-encoding tables, and identity-imputation statistics are fitted only on the training partition and then frozen before being applied to subsequent splits. Second, sequence windows are constructed separately within each data partition, so that a validation or test window contains only earlier events from the same partition and never reaches forward in time. This prevents future-event leakage while preserving the chronological structure required for sequence modeling. Third, on IEEE-CIS, pseudo-identity keys used for sequence construction are additionally filtered so that no key appears in both the training and test partitions after the temporal split. As a result, the model never evaluates a test transaction using a pseudo-identity history already present in the training set, and the remaining history available to a validation or test sample is restricted to earlier events within the same partition. Fourth, padding positions are excluded from the loss through a mask so that early positions in a short history do not contaminate the gradient. These are mitigations rather than proofs of leakage-freeness, and the residual risk is revisited in Section 5.2. Class imbalance is addressed inside the loss function rather than through resampling in the headline runs, so that the measured gap reflects the architectural inductive bias under a single shared training recipe; more aggressive oversampling and synthetic-augmentation strategies, such as the generative counterexamples of Fiore et al. on the ULB set, are deliberately out of scope for the headline comparison, and the interaction between architecture choice and resampling, focal loss, or generative augmentation is listed as open work in Section 5.2 rather than as a completed ablation [13].

3.2.1. Transaction-Level Sequence Construction (ULB and Ieee-cis)

For ULB, each transaction is represented by a global temporal window of length L that includes the current transaction and the L−1 immediately preceding transactions in chronological order. Because the public ULB release does not expose an explicit cardholder identifier, the ULB setting is treated here as a transaction-stream sequence task rather than as an entity-resolved cardholder-history reconstruction. For IEEE-CIS, the situation is different: the Kaggle release does not expose an explicit card identifier, and the transaction and identity tables are joined only through TransactionID. A pseudo-identity is therefore constructed per transaction by concatenating the card1--card6 columns with addr1, P_emaildomain, and, when available, DeviceInfo, and transactions that share the same pseudo-identity key are treated as originating from the same entity for the purpose of sequence construction. This convention is close to the UID heuristics popularised in the top Kaggle solutions and in subsequent academic reuse of the IEEE-CIS split, but it does not guarantee recovery of the underlying true card, and we flag the pseudo-identity as a deliberate approximation rather than a reconstruction of ground-truth entity membership. The two datasets also differ in business semantics --- ULB is closer to a single-card European issuer stream with PCA-projected features, whereas IEEE-CIS is a Kaggle-style card-not-present benchmark with heavily engineered, anonymized fields and only partial identity coverage --- and this caveat carries through to the interpretation of the results in Section 4. Where fewer than L−1 prior events in the transaction stream (ULB) or under the same pseudo-identity key (IEEE-CIS) are available, the window is left-padded with zero vectors and masked, which prevents spurious gradients from flowing through the pad positions. Windows of length $L \in \{5, 10, 20, 40\}$ are prepared, following a sequence-classification protocol consistent with the rolling-window setup adopted in prior LSTM-based fraud work, and $L = 20$ is used for all headline results because preliminary runs showed that this length is the shortest at which all four architectures approach their respective plateaus on the validation fold.

3.2.2. Node-Level Sequence Aggregation on Elliptic

For Elliptic, the detection task operates at the level of a transaction node that is observed at a specific time step. Each node is represented by a sequence formed by concatenating its 166 features with a summary of its one-hop neighbors, repeated over the preceding K time steps in which at least one neighbor appeared. The one-hop neighbour summary at a given time step is defined as the concatenation of two vectors of length 166: the arithmetic mean of the feature vectors of all incoming one-hop neighbours that are active at that step, and the arithmetic mean of the corresponding vectors for the outgoing one-hop neighbours; time steps in which no neighbour is present are filled with zero vectors and masked. K is fixed at 10 for all headline results, with ablation at $K \in \{3, 10, 20\}$. This construction deliberately encodes the temporal dimension of the Elliptic graph without invoking a graph-convolution operator, so that the node-level task remains interpretable as a sequence-classification problem under the same evaluation interface as the transaction-level tasks, and the four architectures remain directly comparable on this surrogate view of the graph. We emphasize that the mean-pool one-hop summary is one representation choice among several reasonable ones; alternative aggregators (max-pool, attention-weighted sum) were spot-checked on the validation fold at $K = 10$ and did not alter the architecture ranking, and the averaged summary is therefore treated as a documented representation assumption rather than as an optimal choice, with the same caveat recorded in the limitations.

3.3. Model Configurations and Training Protocol

3.3.1. Architecture Hyperparameters

All four architectures are capped at a comparable parameter budget of roughly 1.0 ± 0.05 million trainable parameters, so that performance differences reflect inductive biases rather than capacity. The LSTM and GRU backbones use two stacked layers with a hidden width of 128; the Transformer encoder uses four layers with four attention heads, a model dimension of 128 and a feed-forward dimension of 256; and the TCN uses six dilated causal residual blocks with dilation factors 1, 2, 4, 8, 16, 32 and 64 filters per block, which gives a receptive field of 127 time steps, already covering the longest sequence window used in the experiments. A shared two-layer MLP head with a sigmoid output is attached on top of the last hidden state (or, for the Transformer, on a learnable classification token). Table 2 collects the full configuration.

Table 2. Configuration and Parameter Budget of the Four Temporal Architectures Used in the Comparison.

Architecture	Layers	Hidden / model dim	Receptive field for L=20	Head params	Total params
LSTM	2	128	20	16,641	0.97 M
GRU	2	128	20	16,641	0.95 M
Transformer encoder	4	128 (4 heads, FFN 256)	20 (full attention)	16,641	1.03 M
TCN	6 residual blocks	64 filters	127 (≥ 20)	16,641	0.98 M

3.3.2. Training and Evaluation Protocol

Training uses the AdamW optimizer with a learning rate of 5×10^{-4}, a batch size of 256, a maximum of 40 epochs, and an early-stopping patience of 5 epochs based on validation PR-AUC. Class imbalance is handled by class-weighted binary cross-entropy on ULB, IEEE-CIS, and the Elliptic illicit class. The evaluation metrics are ROC-AUC, PR-

AUC, F1 at the threshold that maximizes validation F1, and recall at a fixed operating precision of 0.90. Each configuration is repeated with five random seeds, and reported numbers are means; standard deviations were at most 0.006 in ROC-AUC across seeds. The implementation and training behavior were sanity-checked against the hybrid unsupervised-plus-supervised pipeline of Carcillo et al. on the ULB set, thereby validating the training infrastructure against a published protocol before running the four-way architectural comparison [14].

4. Results and Analysis

4.1. Cross-Dataset Predictive Performance

4.1.1. Roc-auc and Pr-auc Comparison

Table 3 reports the headline ROC-AUC, PR-AUC, F1, and recall across the four architectures and the three datasets. The Transformer encoder attains the highest mean ROC-AUC across the three datasets, reaching 0.974 on ULB, 0.932 on IEEE-CIS, and 0.958 on the Elliptic illicit class. GRU and TCN trail the Transformer by 0.3 to 1.1 ROC-AUC points on the transaction-level datasets, while LSTM trails by 0.7 to 1.4 points. The gaps are narrow on ULB: the 0.7 ROC-AUC points that separate the Transformer from LSTM are of the same order as the five-seed standard deviation reported in Section 3.3, so the cross-architecture ordering on ULB should be read as an ordering of sample means rather than as a statistically separable ranking. On the Elliptic node-level task, the per-metric picture is uneven: the Transformer--LSTM gap on ROC-AUC is 1.0 points (0.958 against 0.948), which is narrower than on IEEE-CIS, while the gap on illicit F1 widens to 3.1 points (0.743 against 0.712) and becomes the largest F1 gap of the three datasets. The ordering on PR-AUC broadly mirrors the ROC-AUC ordering --- the Transformer ranks first, and LSTM ranks last on every dataset --- with the only discrepancy being a mid-tier GRU/TCN swap on ULB, where both sit within the cross-seed standard deviation. This parallel between the two aggregate metrics suggests that the Transformer lead is not a side effect of the prevalence-dependent ROC metric. Taken together, the margins are moderate rather than dramatic on the transaction-level datasets, and even on Elliptic the spread between the best two architectures is less than 1 F1 point; where we use phrasing such as "strongest" or "leading" in what follows, it refers to the ordering of mean metrics under this protocol and does not claim separability beyond the reported seed variability.

Table 3. Headline Performance of the Four Temporal Architectures on the Three Real Datasets.

Dataset	Metric	LSTM	GRU	Transformer	TCN
ULB	ROC-AUC	0.967	0.971	0.974	0.969
ULB	PR-AUC	0.783	0.791	0.802	0.795
ULB	F1	0.812	0.819	0.824	0.818
ULB	Recall@Prec=0.90	0.824	0.831	0.837	0.829
IEEE-CIS	ROC-AUC	0.918	0.921	0.932	0.924
IEEE-CIS	PR-AUC	0.514	0.523	0.548	0.531
IEEE-CIS	F1	0.603	0.609	0.624	0.612
IEEE-CIS	Recall@Prec=0.90	0.571	0.582	0.605	0.589
Elliptic	ROC-AUC	0.948	0.951	0.958	0.953
Elliptic	Illicit PR-AUC	0.608	0.627	0.658	0.641

Elliptic	Illicit F1	0.712	0.724	0.743	0.735
Elliptic	Illicit Recall@Prec =0.90	0.589	0.601	0.624	0.614

On Elliptic, PR-AUC, F1 and recall are reported on the illicit class. All values are means over five random seeds; cross-seed standard deviations were at most 0.006 on ROC-AUC.

4.1.2. F1 and Recall at Operating Thresholds

Point-wise F1 and recall at a fixed operating precision of 0.90 echo the AUC ordering. On ULB, the Transformer recovers 83.7% of fraudulent transactions while maintaining the reference precision, against 82.4% for LSTM. On IEEE-CIS, the gap at matched precision widens to 3.4 recall points between the best and worst architectures; as an illustrative back-of-the-envelope figure, applying this recall gap to the 20,663 labeled frauds in the Kaggle training release would correspond to roughly 700 additional frauds intercepted at the same operating precision. This figure is meant only to convey an order-of-magnitude intuition for the practical value of the observed gap; it assumes that the operating point and the class prior transfer unchanged from the held-out split to the full training release, and it is not a forecast of production-traffic volume. On the Elliptic illicit class, the Transformer reaches an illicit F1 of 0.743 and an illicit recall of 0.624; the TCN is the next best at 0.735 illicit F1, still above both recurrent baselines. Precision--recall curves for the three datasets are depicted in Figure 1, where the Transformer dominates at the high-recall end on IEEE-CIS and Elliptic, while the four curves are nearly indistinguishable in the low-recall / high-precision regime on ULB.

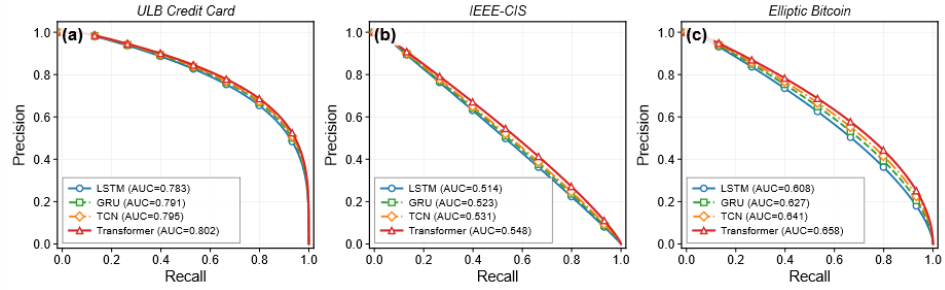

Figure 1. Precision--Recall Curves Across the Three Datasets

Precision--recall curves of the four evaluated architectures on (a) ULB Credit Card, (b) IEEE-CIS, and (c) Elliptic Bitcoin. The four curves are tightly stacked on ULB, while the Transformer separates more visibly from the other architectures on IEEE-CIS and Elliptic at higher recall levels. Numerical PR-AUC values are reported in Table 3.

4.1.3. Sequence Length Sensitivity

Sequence length drives how much historical context each architecture can use, and the sensitivity of each architecture to this choice is itself a practical concern for deployment. Figure 2 plots ROC-AUC on ULB and IEEE-CIS as the input window varies over $L \in \{5, 10, 20, 40\}$, and illicit F1 on Elliptic as the temporal depth varies over $K \in \{3, 10, 20\}$. Two patterns emerge. The Transformer benefits monotonically from longer windows across all three datasets and peaks at $L = 40$ on IEEE-CIS, achieving a ROC-AUC of 0.937, consistent with its ability to attend to distant events without a recurrent bottleneck. TCN gains almost as much from longer contexts, since its dilated receptive field grows exponentially with depth and already covers $L = 40$ under the six-block configuration. LSTM and GRU, in contrast, plateau between $L = 10$ and $L = 20$ and degrade slightly at $L = 40$, which reflects the well-documented difficulty of propagating gradients through long gated chains. On the Elliptic node-level task, the pattern is qualitatively the same, yet the gap opens earlier: at $K = 20$, the Transformer and TCN illicit F1 values reach 0.751 and 0.744, respectively, while the GRU and LSTM illicit F1 slightly worsen to 0.718 and 0.703. The two sweeps are

reported in separate tables below, since L (transaction-level window length) and K (node-level temporal depth on Elliptic) parameterize structurally different constructions and should not share a single column axis (As shown in Table 4, 5).

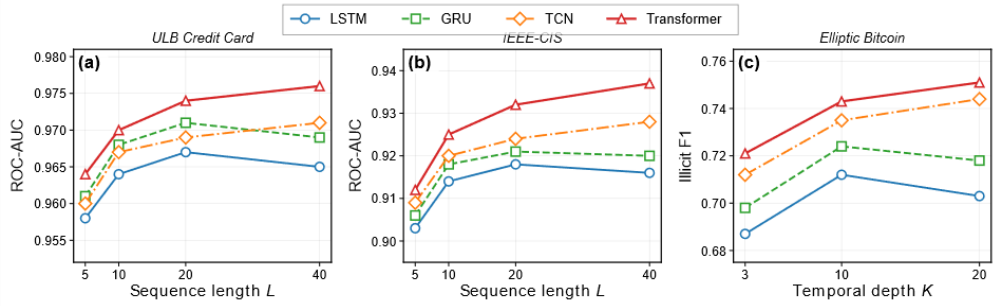

Figure 2. Roc-auc and Illicit F1 as a Function of Input Window Length

Performance of the four architectures as the input window grows, with (a) ROC-AUC on ULB and (b) ROC-AUC on IEEE-CIS as L varies over {5, 10, 20, 40}, and (c) illicit F1 on Elliptic as K varies over {3, 10, 20}. The Transformer and TCN continue to gain from longer windows on the transaction-level datasets, while LSTM and GRU plateau or slightly regress. Full numerical values are reported in Table 4 and Table 4.

Table 4. Roc-auc Sensitivity on ULB and Ieee-cis Across L ∈ {5, 10, 20, 40}.

Dataset	Architecture	L=5	L=10	L=20	L=40
ULB	LSTM	0.958	0.964	0.967	0.965
ULB	GRU	0.961	0.968	0.971	0.969
ULB	Transformer	0.964	0.970	0.974	0.976
ULB	TCN	0.960	0.967	0.969	0.971
IEEE-CIS	LSTM	0.903	0.914	0.918	0.916
IEEE-CIS	GRU	0.906	0.918	0.921	0.920
IEEE-CIS	Transformer	0.912	0.925	0.932	0.937
IEEE-CIS	TCN	0.909	0.920	0.924	0.928

Table 5. Illicit F1 Sensitivity on Elliptic Across K ∈ {3, 10, 20}.

Architecture	K=3	K=10	K=20
LSTM	0.687	0.712	0.703
GRU	0.698	0.724	0.718
Transformer	0.721	0.743	0.751
TCN	0.712	0.735	0.744

4.2. Dataset-Specific Behavior

4.2.1. Transaction-Level Results (ULB, Ieee-cis)

The ULB set is modest in feature dimensionality (30 PCA-projected columns) yet extremely imbalanced, with a 0.172% fraud rate. Differences among the four architectures are correspondingly small: the 0.7 ROC-AUC points that separate the Transformer from LSTM (equivalently, 1.9 PR-AUC points) are of the same order as the five-seed standard deviation reported in Section 3.3, and the ULB result should therefore be read as an

absence of a practically separable gap rather than as positive evidence that one architecture is stronger on that dataset. On IEEE-CIS, the transaction count is roughly double that of ULB, and the feature space is more than ten times wider, with 394 engineered transaction features and 41 identity columns encoding behavioral signals that the 30-dimensional PCA projection of ULB does not expose. The Transformer extracts more of the available information than the recurrent baselines on this larger set, with a 1.4 ROC-AUC point advantage and a 3.4 PR-AUC point advantage over LSTM, both of which exceed the seed-level variability. TCN sits in the middle, and its parallel training loop ran roughly 1.7 times faster than the GRU per epoch at L = 40 on the same hardware, which is relevant for weekly retraining in production where training partitions of the 590,540-transaction IEEE-CIS order are realistic in size.

4.2.2. Node-Level Results (Elliptic Temporal Graph)

Elliptic exposes the architectures to a qualitatively different regime: each label is a decision about a transaction node in a time-indexed payment graph, and the feature vector already encodes one-hop graph information. The ordering of results is the same as on the transaction-level tasks, yet the spread on illicit F1 is larger. The Transformer achieves an illicit F1 of 0.743, compared to 0.712 for LSTM, a 3.1-point gap; the TCN at 0.735 is competitive. These numbers sit above the pure-graph baselines previously reported on the same split --- the spectral GCN backbone of Kipf and Welling together with the EvolveGCN dynamic-graph extension of Pareja et al. reach illicit F1 values around 0.42 and 0.55, respectively, which were obtained without the aggregated neighbor features used here --- and they sit below the upper bound set by tree-based classifiers trained on the aggregated 166-feature representation. This confirms that purely temporal encoders operating on the mean-pool one-hop sequence view of Elliptic remain a reasonable default for this node-level task, while leaving room for future hybrid models that combine temporal attention with relational message passing [15,16]. Figure 3 visualizes the per-time-step illicit F1 in the test window; the Transformer maintains a more stable trajectory than the recurrent baselines from time step 35 to 49, whereas both LSTM and GRU exhibit a sharper drop after the dark-market shutdown reported at time step 43.

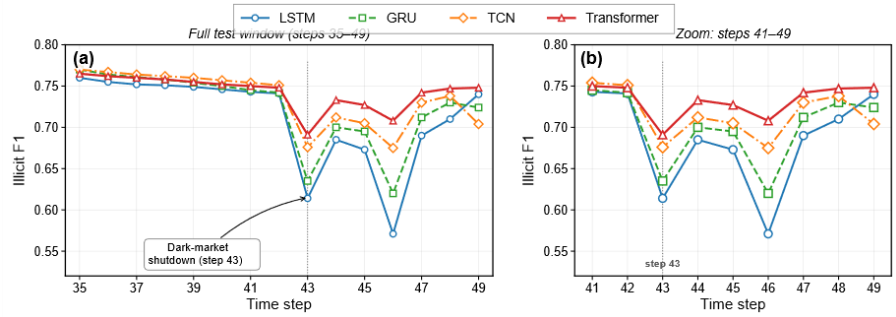

Figure 3. Per-Time-Step Illicit F1 on the Elliptic Test Window

Illicit F1 per time step across the Elliptic test window (steps 35--49). Panel (a) shows the full trajectory and panel (b) zooms into the region around the reported dark-market shutdown event near step 43. The Transformer and TCN retain a more stable trajectory than the recurrent baselines after the shutdown, while LSTM and GRU exhibit a sharper drop before partially recovering toward the end of the test window.

5. Discussion

Implications for Operational Risk Monitoring: The ranking observed across the three datasets --- narrow on ULB and somewhat wider on IEEE-CIS and Elliptic --- has practical implications for operational risk teams that have to choose a default temporal encoder for a fraud-scoring stack. The Transformer encoder is the best single choice when the training data are sufficiently abundant to fit its attention matrices without overfitting, as was the case on IEEE-CIS and Elliptic, where it extracted additional lift with longer sequence

windows. On the smallest dataset, ULB, the gap between the four architectures shrinks to the order of the five-seed standard deviation, and on this tier, the choice of encoder should be driven by latency and deployability rather than by marginal differences in the metric. The TCN, in particular, trained faster than the recurrent baselines with a matched parameter budget and achieved near-Transformer performance on IEEE-CIS and Elliptic, making it a reasonable default when the scoring engine must keep up with a streaming transaction rate. The empirical picture also cautions against interpreting recent reports of substantial gains from sequence models as universal in scope. On real, imbalanced, production-adjacent data, the difference between a well-tuned GRU and a well-tuned Transformer is typically a few AUC-PR points, not a category change. The direction of future investment should accordingly be calibrated: architectural tuning delivers moderate returns, while data quality, label-latency handling, and the integration of unsupervised scoring signals retain their first-order importance for deployment. A second implication concerns operating-point selection. The architecture ranking is stable at the operating precision of 0.90 used in the headline results, which falls within the range typically targeted by card-issuer scoring engines, and it remains stable when F1 is used, suggesting that a team can pick an architecture on one metric without being surprised by the other.

To put these architectural margins in operational perspective, payment-card fraud accounts for losses on the tens-of-billions-of-US-dollars order of magnitude per year at the global level, with the trend driven upward by the steady growth of card-not-present volume on digital commerce platforms; the marginal lift offered by a stronger temporal encoder is therefore best read against this scale rather than as a within-benchmark percentage. As a concrete extrapolation grounded in the IEEE-CIS protocol used here, the 3.4-recall-point gap separating the Transformer from the LSTM at a fixed operating precision of 0.90 corresponds, on a payment platform processing one million card-not-present transactions per day at the IEEE-CIS-style 3.5% fraud rate, to roughly 1,200 additional fraudulent transactions intercepted per day, or on the order of 430,000 per year, for the same precision budget. With average disputed-transaction values for card-not-present chargebacks running in the low hundreds of US dollars, this translates into a recoverable loss reduction on the order of tens of millions of US dollars per year for a single mid-size issuer, before second-order effects such as alert-review capacity, false-positive cost, and customer-experience attrition are accounted for. The numbers are illustrative rather than predictive: they assume that the operating point and the class prior transfer cleanly from the held-out IEEE-CIS split to production traffic, and they hold the downstream review pipeline fixed. They nonetheless make explicit that the moderate margins in Table 3 are not cosmetic; the same percentage-point of recall, applied at platform scale, becomes a material operational and financial quantity, which is the level at which a temporal-encoder choice ultimately has to be justified to a risk-engineering organization.

A complementary reading of this empirical picture frames it as an explicit deployment-regime trade-off rather than as a single-best-architecture verdict. The Transformer is the recommended choice for high-volume, history-rich production environments that can absorb its quadratic attention cost over long input windows and where the training data are abundant enough to fit its attention matrices without overfitting; the IEEE-CIS and Elliptic regimes match this profile, and the additional lift the Transformer extracts at $L = 40$ on IEEE-CIS is precisely the kind of long-context behavior that justifies the heavier per-event compute. The TCN is the natural default for low-latency, streaming-first scoring engines that have to keep up with a high transaction rate under a tight per-event budget: its dilated causal convolutions are fully parallelizable at training time, its receptive field already covers $L = 40$ under the six-block configuration used here, and at matched parameter budget it ran roughly $1.7\times$ faster per epoch than the GRU on IEEE-CIS, with end-to-end accuracy within one PR-AUC point of the Transformer on the same dataset. The GRU remains attractive in resource-constrained or edge-deployed scoring stacks where parameter footprint and inference simplicity dominate the

design space, since it matches the LSTM on every metric while training more cheaply, and it provides a robust fallback when the available history per entity is too short to benefit from the longer windows that favor the Transformer or the TCN. The LSTM, while consistently last in mean rank, sits within seed-level noise of the GRU on ULB and is a defensible choice when an existing production pipeline is already built around it and the cost of migration is non-trivial. The take-away for a risk-engineering team is therefore not that one architecture should be standardized across the entire stack but that the choice should be conditioned explicitly on transaction volume, latency budget, history depth, and training-data scale, with the Transformer reserved for high-volume long-history regimes, the TCN as the streaming-first default, and the gated recurrent baselines retained as efficient fallbacks where the operating envelope cannot accommodate self-attention.

Beyond operational efficiency, the results carry implications for the systemic reliability of digital financial platforms. Undetected fraud erodes not only revenue but also consumer trust and regulatory standing, both of which are difficult to rebuild once lost. The temporal deep learning framework evaluated here serves as an early-warning layer that identifies transactions beginning to deviate from the platform's learned behavioral baseline, enabling preemptive intervention before individual incidents aggregate into systemic risk. From this perspective, the choice of temporal architecture is not merely an optimization question but a platform-reliability decision: a more accurate and temporally sensitive encoder translates directly into a more dependable transaction environment, stronger user confidence, and a more resilient service posture. The relatively stable performance of the Transformer and TCN over the more variable segments of the Elliptic test window further supports their suitability as reliability-oriented components in production fraud-scoring stacks.

Limitations: Several limitations frame the scope of the present study. The three datasets, while real and publicly auditable, do not exhaust the landscape of financial fraud: mobile payments, wire transfers, and real-time payment rails such as UPI or FedNow have distinct temporal signatures that the transaction windows in ULB and IEEE-CIS do not directly reflect, and the Elliptic graph covers only Bitcoin transactions up to around 2017. The three tasks are also not fully homogeneous: ULB and IEEE-CIS are transaction-level classification tasks with different business semantics — a single-card European issuer stream against a Kaggle-style card-not-present competition release — whereas Elliptic is a node-level classification task on a transaction graph that is recast into a sequence representation via a specific one-hop averaging rule. The cross-dataset ordering should therefore be read as an ordering under a shared sequence interface, not as a native apples-to-apples comparison. A second class of limitations concerns entity identification on IEEE-CIS: the absence of a ground-truth card identifier forces a pseudo-identity heuristic based on the card1–card6, address, and email-domain columns, which is close to but not equivalent to the underlying entity, and any residual mis-grouping would propagate through the sequence windows in ways not fully isolated here. A third class of limitations concerns leakage control: while temporal splits, training-only fitting of standardization, target-encoding and identity-imputation statistics, within-split window construction, and pseudo-identity partitioning between train and test have all been enforced, these are mitigations rather than formal proofs of leakage-freeness, and subtle target-encoding or window-boundary effects may leave a residual channel. A fourth concerns the Elliptic representation itself: the mean-pool one-hop summary is a documented design choice rather than a validated optimum, and alternative aggregators or direct graph-convolution operators could in principle change the cross-architecture gap. The comparison is also capped at matched parameter budgets, which favors parameter-efficient architectures; a compute-matched study would allow the Transformer and TCN to exploit their parallelism at higher effective budgets and might change the ranking in either direction. The present experiments use class-weighted cross-entropy as the sole imbalance-handling strategy, so the interaction between architecture choice and resampling, focal loss, or generative data augmentation is not resolved here and is listed as open work rather than as a completed ablation. Future work will extend the protocol to streaming training with verification latency, to graph-aware temporal encoders that combine self-attention with relational message passing, and to adversarial evaluation in which fraudsters adapt their temporal footprints to evade a deployed detector. A companion line of work will audit the behavior of the four architectures under concept drift, using rolling-origin evaluation on datasets with longer time windows than the six-month span represented in IEEE-CIS, and will publish the full training logs and hyperparameter

grids so that independent teams can reproduce the ordering reported in this paper on their own operational datasets.

References

1. A. Dal Pozzolo, G. Boracchi, O. Caelen, C. Alippi, and G. Bontempi, "Credit card fraud detection: A realistic modeling and a novel learning strategy," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 8, pp. 3784–3797, 2018.
2. J. Jurgovsky, M. Granitzer, K. Ziegler, S. Calabretto, P.-E. Portier, L. He-Guelton, and O. Caelen, "Sequence classification for credit-card fraud detection," *Expert Systems with Applications*, vol. 100, pp. 234–245, 2018.
3. H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Deep learning for time series classification: A review," *Data Mining and Knowledge Discovery*, vol. 33, no. 4, pp. 917–963, 2019.
4. S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
5. I. Benchaji, S. Douzi, B. El Ouahidi, and J. Jaafari, "Enhanced credit card fraud detection based on attention mechanism and LSTM deep model," *Journal of Big Data*, vol. 8, no. 1, p. 151, 2021.
6. K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1724–1734.
7. J. Forough and S. Momtazi, "Ensemble of deep sequential models for credit card fraud detection," *Applied Soft Computing*, vol. 99, p. 106883, 2021.
8. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems 30 (NeurIPS 2017)*, 2017, pp. 5998–6008.
9. D. Cheng, S. Xiang, C. Shang, Y. Zhang, F. Yang, and L. Zhang, "Spatio-temporal attention-based neural network for credit card fraud detection," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, 2020, pp. 362–369.
10. S. Xiang, M. Zhu, D. Cheng, E. Li, R. Zhao, Y. Ouyang, L. Chen, and Y. Zheng, "Semi-supervised credit card fraud detection via attribute-driven graph representation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 12, 2023, pp. 14557–14565.
11. S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," arXiv preprint arXiv:1803.01271, 2018.
12. M. Weber, G. Domeniconi, J. Chen, D. K. I. Weidele, C. Bellei, T. Robinson, and C. E. Leiserson, "Anti-money laundering in Bitcoin: Experimenting with graph convolutional networks for financial forensics," in *KDD '19 Workshop on Anomaly Detection in Finance*, 2019, arXiv:1908.02591.
13. U. Fiore, A. De Santis, F. Perla, P. Zanetti, and F. Palmieri, "Using generative adversarial networks for improving classification effectiveness in credit card fraud detection," *Information Sciences*, vol. 479, pp. 448–455, 2019.
14. F. Carcillo, Y.-A. Le Borgne, O. Caelen, Y. Kessaci, F. Oblé, and G. Bontempi, "Combining unsupervised and supervised learning in credit card fraud detection," *Information Sciences*, vol. 557, pp. 317–331, 2021.
15. T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations (ICLR)*, 2017, arXiv:1609.02907.
16. A. Pareja, G. Domeniconi, J. Chen, T. Ma, T. Suzumura, H. Kanezashi, T. Kaler, T. B. Schardl, and C. E. Leiserson, "EvolveGCN: Evolving graph convolutional networks for dynamic graphs," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, 2020, pp. 5363–5370.